HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
COMPUTER SCIENCE & ENGINEERING

Computer Arichtecture

Lab 6

Chapter 4 : The Processor

Nguyễn Hữu Hiếu - 2013149

Ho Chi Minh City, 05/2022

Contents

Bài 1: Xác định clock rate

Cho thời gian delay của các khối như Bảng 1

Bảng 1: delay của các khối phần cứng

Phần cứng	Delay(ns)
Instruction memory	150
Register	100
ALU	100
Data memory	150
Các bộ phận cứng khác	0

Xét đoạn chương trình sau:

```
1         addi $t1, $zero, 100
2         addi $t2, $zero, 0
3   loop:
4         beq  $t1, $t2, exit
5         addi $t1, $t1, -1
6         addi $t2, $t2, 1
7         j loop
```

(a) Xác định clock cycle của hệ thống single clock, multi clock và pineline clock.

- *Single clock rate = thời gian thực thi lệnh dài nhất (lệnh load)*

  = I-Mem + Register + ALU + D-Mem
  = 150 + 100 + 100 +150 = 500ns.

- *Multiple clock rate = max của khối có độ delay cao nhất = 150ns.*
- *Pineline clock cycle = max của khối có độ delay cao nhất = 150ns*

(b) Xác định thuời gian thwucj thi của chương trình trên khi chạy với hệ thống single cycle, multi cycle và pineline cycle (không xét stall).

- *Loop lặp lại 50 lần + 1 lần lệnh beq để thoát vòng lặp*

*→ Tổng số lệnh thực thi của chương trình là = 4 * 50 + 1 + 2 = 203 lệnh.*

$CPI_{single\ cycle} = 1$

$CPI_{Multiple\ cycle} = \frac{2}{3} \times 4 + \frac{1}{6} \times 3 + \frac{1}{6} \times 2 = 3.5$

$CPI_{ideal\ of\ Pineline} = 1$

Vậy thời gian thực thi của chương trình là:

$Time_{single\ cycle} = 1 \times 203 \times 500 = 101500ns.$

$Time_{Multiple\ cycle} = 3.5 \times 203 \times 150 = 106578ns.$

$Tim_{Pineline} = 1 \times (203 + 5 - 1) \times 150 = 31050ns.$

(c) Tính speed up của hệ thống pineline với hệ thống multi cycle và single cycle.

$$Speedup = \frac{101500}{31050} = 3.27$$

(d) Khi delay ALU thay đổi từ 100 → 150. Tính lại kết quả câu a,b,c.

- Câu a.
  - *Single clock rate = 550ns.*
  - *Multiple clock rate = 200ns.*
  - *Pineline clock cycle = 200ns.*
- Câu b.
  - $Time_{single\ cycle} = 1 \times 203 \times 550 = 111650ns.$
  - $Time_{Multiple\ cycle} = 3.5 \times 203 \times 200 = 142100ns.$
  - $Tim_{Pineline} = 1 \times (203 + 5 - 1) \times 200 = 41400ns.$
- Câu c.

$$Speedup = \frac{111650}{41400} = 2.7$$

Bài 2: Xử lý Hazard

Dùng lại đoạn code của Bài 1:

(a) Xác định sự phụ thuộc dữ liệu trong đoạn chương trình trên.

- *Sự phụ thuốc dữ liệu : Read After Write – RAW Hazard*
- *Trước vòng lặp – lệnh (3) phụ thuộc (1) và (2).*
- *Sau vòng lặp – lệnh (3) phụ thuộc lệnh (5)*

(b) Giải quyết data hazard bằng chèn stall (giải quyết bằng phần mềm), khi thực thi đoạn code trên với hệ thống pipeline thì cần chèn vào bao nhiêu stall (khựng lại)?

- *Chèn stall: nop ở sau lệnh số (2) và nop ở ngay nhãn loop.*

(c) Dùng cơ chế forward để giải quyết hazard (giải quyết bằng phần cứng), khi đó có bao nhiêu stall? Vẽ hình minh họa.

(d) Dùng cơ chế forward, stall để giải quyết hazard (control và data), khi đó có bao nhiêu stall?

- *Không còn stall.*

(e) Ngoài 2 cơ chế trên, ta có thể giảm stall bằng cánh sắp xếp lại thứ code (giải quyết bằng trình biên dịch compiler). Hãy sắp xếp lại code sao cho ít stall nhất.

- *Với đoạn code trên, không thể sắp xếp lại code vì sự phụ thuộc tuyến tính của dòng trên và dòng dưới.*

Bài 3: Xử lý hazard (lệnh load)

Cho đoạn code sau:

```
1 addi $t1, $zero, 100
2 addi $t2, $zero, 100
3 add  $t3, $t1,   $t2
4 lw   $t4, 0($a0)
5 lw   $t5, 4($a0)
6 and  $t6, $t4, $t5
7 sw   $t6, 8($a0)
```

Trả lời câu hỏi trong Bài 2.

(a) Xác định sự phụ thuộc dữ liệu trong đoạn chương trình trên.

- *Sự phụ thuộc dữ liệu: Read After Write – RAW Hazard*
  Lệnh (3) phụ thuộc lệnh (2) và (1).
  Lệnh (6) phụ thuộc lệnh (5) và (4).
  Lệnh (7) phụ thuộc lệnh (6).

(b) Giải quyết data hazard bằng chèn stall (giải quyết bằng phần mềm), khi thực thi đoạn code trên với hệ thống pipeline thì cần chèn vào bao nhiêu stall (khựng lại)?

- *Ta cần chèn 6 stall*
  2 stall giữa (2) và (3).
  2 stall giữa (5) và (6).
  2 stall giữa (6) và (7).

(c) Dùng cơ chế forward để giải quyết hazard (giải quyết bằng phần cứng), khi đó có bao nhiêu stall? Vẽ hình minh họa.

- *Còn 1 stall giữa (5) và (6)*

(d) Dùng cơ chế forward, stall, để giải quyết hazard (control và data), khi đó có bao nhiêu stall?

- *Còn 1 stall giữa lệnh (5) và (6)*

(e) Ngoài 2 cơ chế ở trên, ta có thể giảm stall bằng cách sắp xếp lại thứ tự code (giải quyết bằng trình biên dịch compiler). Hãy sắp xếp lại code sao cho ít stall nhất.

- *Một trong những cách sắp xếp giảm stall*

(4)→ (5) → (1) → (2) → (6) → (3) → (7)

```
1 lw   $t4, 0($a0)       #4
2 lw   $t5, 4($a0)       #5
3 addi $t1, $zero, 100   #1
4 addi $t2, $zero, 100   #2
5 add  $t6, $t4, $t5     #6
6 add  $t3, $t1, $t2     #3
7 sw   $t6, 8($a0)       #7
```

Còn lại một stall giữa (6) và (3)